Eurographics Conference on Visualization (EuroVis) 2016
E. Bertini, N. Elmqvist, and T. Wischgoll
(Guest Editors)

Short Paper

Visualizing Flow Fields Using Fractal Dimensions

Han-Wei Shen, Ross Vasko, and Rephael Wenger

The Ohio State University, Columbus, OH, USA

Abstract

Streamlines are a popular way of visualizing flow in vector fields. A major challenge in flow field visualization is selecting the streamlines to view. Rendering too many streamlines clutters the visualization and makes features of the field difficult to identify. Rendering too few streamlines causes viewers to completely miss features of the flow field not rendered.
The fractal dimension of a streamline represents its space-filling properties. To identify complex or interesting streamlines, we build a regular grid of scalar values which represent the fractal dimension of streamlines around each grid vertex. Vortices and turbulent regions are often associated with regions of high fractal dimension. We use this scalar grid both to filter streamlines by fractal dimension and to identify and visualize regions containing vortices and turbulence. We describe an interactive tool which allows for quick streamline selection and visualization of regions containing vortices and turbulence.

Categories and Subject Descriptors (according to ACM CCS): I.3.0 [Computer Graphics]: General—Flow Field Visualization

1. Introduction

A flow field is a specific type of vector field that represents the flow of some fluid. Each point in a flow field is a vector that represents the direction and rate of mass transport of a flow. Flow fields are used in several different fields of science and engineering. They can be used to model large scale simulations such as atmospheric behavior, air flow during wind tunnel tests, or blood flow. Flow field data sets contain several different features, such as vortices, and have regions with different types of flow behavior. Visualizing these different regions and features is crucial to understanding the flow data.

A common way to visualize flow fields is with streamlines. A streamline is a curve that is tangent to the velocity vector of the flow field at each point of the curve. Streamlines are computationally inexpensive to generate and allow a viewer to see the behavior of a region of the flow field. A large challenge in using streamlines to visualize flow fields is choosing which streamlines to display to both prevent cluttering the visualization as well as ensuring a proper sampling of the flow field.

To appropriately filter the streamlines and identify important flow field features, we examine the geometric properties of streamlines using the box counting ratio defined by Khoury and Wenger. The box counting ratio measures the space-filling properties of an object and quantifies its complexity. With this measurement, we are then able to categorize streamlines based on their complexity. Such a measurement allows for filtering to remove clutter while still retaining important or defining flow field features. We construct a scalar grid with values that are representative of the complexity of the streamlines in some nearby neighborhood. This scalar field allows us to easily identify which regions of the flow field are complex. Lastly, we apply a variety of interactive visualization techniques to the scalar field to allow the user different insights on the flow data.

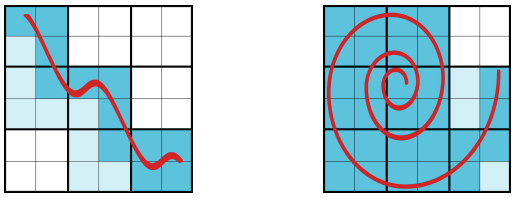
2. Related Work

There has already been a considerable amount of work done in streamline filtering and flow field feature identification, but many of these methods require prerequisite knowledge of the flow field or depend on restrictive definitions of features. Streamline seeding in 2D or on surfaces in 3D is discussed in [TB96], [MHHI98], and [MAD05]. While these techniques create clear and evenly spaced streamline visualizations for surfaces, they are difficult to extend into a general 3D space.

McLoughlin et al. [MJL*13] seek to reduce streamline clutter by requiring a predefined rake and then removing streamlines along that rake that exhibit significant amounts of similarity. Viewpoints of streamlines are evaluated in [MCHM10] by analyzing the amount of occlusion present. Streamline filtering methods based on concepts of entropy are discussed in [XLS10] and [LMSC11].

Critical points of the flow field are analyzed in [YKP05] and different seeding strategies are used depending on the behavior near the critical points. [JBTS08], [Jän10], and [BKH*15] use concepts such as statistical complexity and invariant moments to indentify various regions of flow fields. In [SS06], various definitions are constructed for different flow field features and the features are searched for in the flow field. Similarly, various features such as

DOI: 10.2312/eurovisshort.20161156



(a) simple line (b) complex line

Figure 1: Examples of the box counting ratio measuring streamline complexity in 2D. (a) Simple line calculated to have a box counting ratio of $\log_2 \frac{12}{5} = 1.26$. (b) Complex line calculated to have a box counting ratio of $\log_2 \frac{29}{8} = 1.86$.

vortices are defined in [HEWK03] and the vectors of the flow field are examined to see if they meet requirements of the properties. In all [SP99], [SPM*98], and [ZHA98] features of a vortex are defined and then searched for in the flow field. [MBS*04] segments the flow field into regions of different behavior.

We attempt to avoid any strict definition of flow field features to prevent restricting our algorithm to only find specific features. Instead, we treat the complexity of a streamline as how it fills a space. In a previous work by Chaudhuri et al. [CLSW14], they introduce measuring the streamline complexity with fractal dimensions to observe behavior at different scales. Fractal dimensions are further discussed in [KW10]. Features in the flow field can then be organized by their complexity as well as what scale the feature appears in. Using the box counting ratio, we are able to both have a general definition of complexity and provide scalar values that are representative of the flow field complexity.

3. Box Counting Ratio

Khoury and Wenger [KW10] defined the box counting ratio and used it to analyze isosurfaces. The box counting ratio of a set is a real number between 0 and 3 that is determined by how the set fills the space. The box counting ratio of a set S is defined as:

$$dim_\varepsilon(S) = \log_2 \frac{N_\varepsilon(S)}{N_{2\varepsilon}(S)} \quad (1)$$

where $N_x(S)$ is the number of boxes that the set S will intersect on a fixed grid that has boxes with edge lengths of x.

We then measure the box counting ratio of the streamlines by defining a fixed grid, counting the number of boxes of widths 2ε and ε that a streamline intersects, and then solving for the final box counting ratio. With this new formula, we expect that the straight and simple streamlines will have a box counting ratio near 1, as they do not have any space filling properties. As the streamline becomes more complex and fills a 3D region more densely, we expect the box counting ratio to increase towards 3. These measurements are illustrated in 2D examples in Figure 1.

4. Streamline Complexity Grid

The streamline complexity grid is a scalar grid which represents the complexity of the flow behavior around each grid vertex using the local box counting ratio. We let 2λ denote the length of each grid edge.

Let ζ be a streamline generated from the flow field and let G_p be a $w \times w \times w$ grid of cubes with edge length 2λ centered at point p. The local box counting ratio of ζ at p is $dim_\lambda(\zeta \cap G_p)$. If the value $N_{2\lambda}(\zeta \cap G_p)$ is below a threshold c, then we discard this measurement. We implement this threshold to ensure that a large enough portion of a streamline is being examined to calculate a stable measurement. In particular, this threshold helps increase the stability of the box counting ratio near the grid boundary.

To generate the values for the streamline complexity grid, streamlines must be generated to properly sample the vector field. We generate this set of streamlines as follows. For each voxel v, generate a new streamline ζ through the center, if v is intersected by fewer than five streamlines, and then determine the voxels intersected by ζ. Each voxel v is intersected by a set of streamlines. We associate with v the first five streamlines (ordered by generation) that intersect v. To calculate the final value $\phi(p)$, first take the streamlines associated with v_p and calculate the local box counting ratio of each streamline at p. Record the median value of these ratios and denote the streamline generating this median value as p_ζ. Finally, average the value at each point with the values of the 26 neighboring points. The resulting values form the streamline complexity grid.

5. Visualization Methods

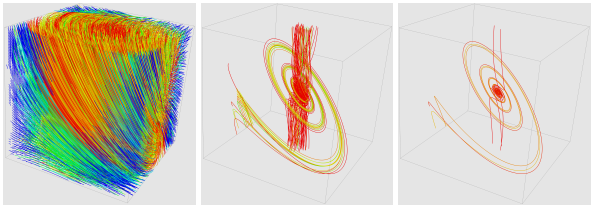
Once constructed, the streamline complexity grid can be used in visualization techniques. We describe the different techniques that are possible with the streamline complexity grid.

5.1. Streamline filtering by value

To remove streamlines in the flow field and to reduce clutter, the user can filter streamlines by their measured complexity values. The user is able to choose two values, a and b where $a < b$, and only display streamlines from voxels with a complexity between the chosen values. The streamline p_ζ will be displayed if $a \leq \phi(p) \leq b$. By choosing values near 1, streamlines with a low box counting ratio and smooth flow will be displayed. By choosing values above 1.4, streamlines will begin to be filtered and the more complex regions of the flow field are highlighted.

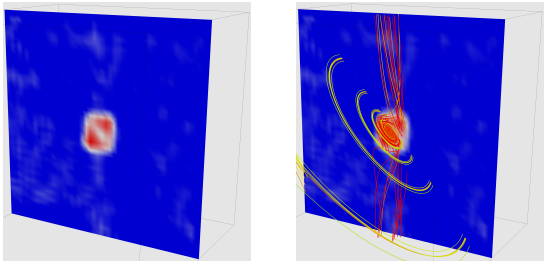
5.2. Local complexity maximums

A significant amount of clutter in the streamline display will remain if additional filtering methods are not considered. Several streamlines in the visualization will be visually similar or provide redundant information. In addition to using a threshold, we can choose voxels whose complexity values are local maxima and render a representative streamline for each such voxel. Local maximum filtering will show streamline p_ζ at point p only if for each q, where q is one of the 8 neighbors of p, $\phi(p) > \phi(q)$. This method allows for single streamline representatives to be shown for each feature or region rather than several, cluttered streamlines. Filtering by complexity value is shown in Figure 2.



(a) (b) (c)

Figure 2: Example of the streamline filtering techniques. (a) The cluttered view of all 5000 streamlines generated. (b) 110 streamlines displayed after filtering. (c) Streamlines filtered by local maximums to show 8 streamlines. Changing this threshold towards the higher complexity values takes approximately 80 ms to update the streamlines shown and then redraw the image.



(a) (b)

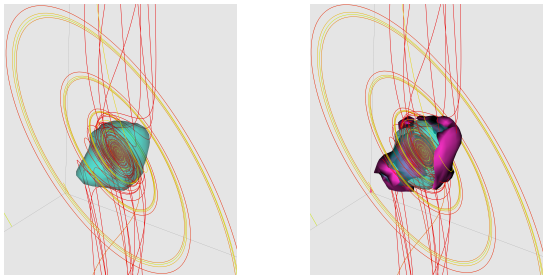
Figure 3: Example visualizations using the colored plane. (a) The plane indicating the regions of high complexity. (b) The high complexity lines near the defined plane. Changing a coordinate of the plane takes approximately 250 ms to update the streamlines shown and to redraw the image.

5.3. Complexity plane

A colored plane can be used to allow the user to visualize the scalar complexity grid, ϕ, directly. A color gradient from blue to white to red is able to be defined and mapped to values in the range 0 to 3, for each of the possible box counting ratios. Low scalar values will be displayed as blue colors, while high scalar values will be displayed as red colors. A plane is then defined on the scalar complexity grid and each point on the plane is colored from this defined color gradient. The user is able to control the plane through the scalar complexity grid to identify regions of varying complexity in the grid. Once regions of interest are identified through the colored plane, the user can display streamlines near that region to understand its behavior. The user is able to view different ratios of both high complexity and low complexity streamlines seeded from the plane. An example of the plane visualization is shown in Figure 3.

5.4. Isosurfaces:

An isosurface $\{x \mid \phi(x) = \sigma\}$ can be used to highlight regions of the flow field with a high complexity. This isosurface will enclose the regions of streamlines with a box counting ratio higher than the σ value and provide a simple way to identify regions of a defined complexity. At particularly high σ values, the isosurface will enclose complex features of the flow field that the user may have otherwise missed.



(a) (b)

Figure 4: Example visualizations with isosurfaces. (b) An isosurface of ϕ enclosing high complexity regions. (b) An isosurface of ϕ_g enclosing regions of high complexity change.

5.5. Streamline gradient magnitudes:

The gradient magnitudes of the streamline complexity grid can be calculated to create a new gradient magnitude scalar grid ϕ_g. The scalar $\phi_g(x)$ is given by $\|\nabla\phi(x)\|$. Another isosurface can be used to visualize the function ϕ_g and to identify regions of high change of complexity. Vortices in the flow field tend to have high complexity values recorded near their centers, with values quickly decreasing towards their boundaries. When a high isovalue is chosen for the gradient magnitude isosurface, the isosurface will often highlight these isolated regions of turbulence or turbulent regions that quickly become smooth. Examples of isosurfaces of the scalar complexity values and gradient magnitudes are shown in Figure 4.

6. Results

The algorithm was implemented in C++ using The Visualization Toolkit (VTK) across two separate programs. The first program generates the streamline complexity information and the second program allows real-time interaction with the data set.

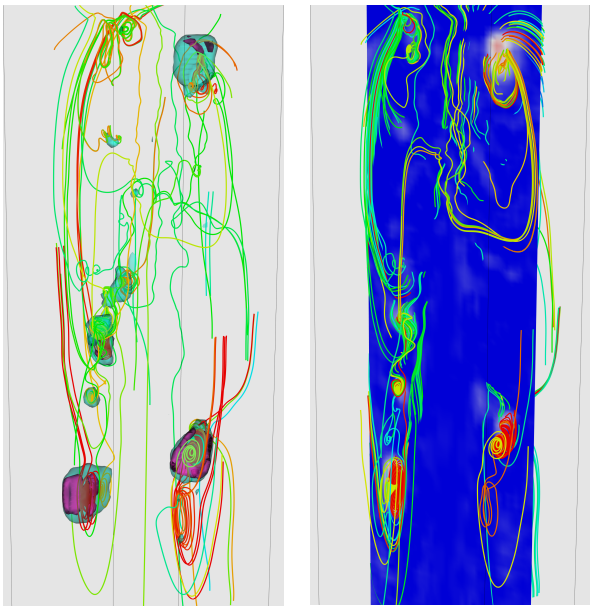
6.1. Synthetic data sets

We used synthetic data sets with known regions of high complexity to verify the correctness of our algorithm. In the synthetic data sets generated and used, we were successfully able to capture the high complexity regions. Such a data set is shown in examples in Section 5.

6.2. Natural data sets

The algorithm was also used to extract the high complexity regions from natural data sets. We mainly used the Solar Plume data set (courtesy of NCAR) and the Hurricane Isabel data set, which was the data set in the 2005 IEEE Visualization contest.

The Solar Plume data set is a $126 \times 126 \times 512$ vector field. This data set has many different regions of varying flow complexity. Central regions of the flow tend to be more laminar with vortices near the boundaries of the data set.



(a) Solar Plume filtered (b) Solar Plume with plane

Figure 5: Example of how filtering increases the visibility of flow field features in the Solar Plume data set. (a) The set of streamlines filtered by complexity measurements from an original data set. (b) The plane allows the viewer to see the different regions of high and low complexity.

A filtering of the streamlines produced by considering the complexity measurements given by our algorithm along with isosurfaces is displayed in Figure 5.a. The streamlines are also colored by complexity, with the red streamlines being the most complex and often contain vortices and green streamlines being less complex. The aqua isosurface in this rendering is set to a value near 1.2 to highlight the complex regions of the flow field. Additionally, the purple isosurface is the gradient isosurfaces and highlights regions around vortices that have a high change in complexity. A plane with nearby streamlines is shown in Figure 5.b. The colored plane highlights the vortices near the boundaries of the data set. Rather than manually attempting to identify which regions of the flow field exhibit specific types of behavior, this visualization allows the viewer to identify these regions quickly.

This Solar Plume data set had a λ parameter of 1.5 and a w parameter of 8. It contains 21,000 streamlines and took approximately 184 seconds to generate. The viewer program takes approximately 17 seconds to preprocess the data set and display an initial visualization.

We found similar results with the Hurricane Isabel data set. Our algorithm was able to successfully identify regions of high complexity and provided a simple way for viewers to find vortices.

6.3. Dependence on parameters

The parameters that have the most significant effect on the output streamline complexity grid are the length of the grid edges, 2λ, and the size of the window used in the local box counting ratio calculations, w.

As the λ parameter decreases, we are able to obtain a much higher resolution streamline complexity grid and identify finer features of the flow field. However, this increase in resolution creates a significant increase in running time and storage space. This is due to the significant increases in streamlines needed to be generated and the number of box intersections needed to be calculated. We typically try to choose a λ parameter so that the streamline complexity grid matches the resolution of the flow field.

The w parameter has an influence on the accuracy of the box counting ratio calculations and the size of the features captured. On the one hand, we would like to use a large window to decrease the effects of the fixed grid alignment and to create a more stable measurement. On the other hand, a high w parameter may cause the box counting ratio measurements to include multiple features along the streamlines and distort measurements. We choose a w parameter so that the local box counting measurements only consider the feature size present in the data. In our data sets, the w parameter was typically set to 6 or 8.

Although these parameters affect the individual values of streamline complexities, we find that there is a parameter range in which we are able to generate streamline complexity grids with similar regions of high complexity. To show this, we generated data sets with varying parameters and found that the highest complexity regions stayed consistent from data set to data set.

6.4. Limitations

The most significant limitation encountered using the box counting ratio to measure streamline complexity was the alignment artifacts due to small sample sizes. When approximating the fractal dimension we should have very large sample sizes. However, our sample size is limited because λ cannot be set too small and w cannot be set too large. This limited sample size can occasionally cause issues in which the lines complexities are not properly estimated.

Additionally, the box counting ratio is not actually measuring properties of turbulence or vortices. While the box counting ratio is often able to identify regions that contain this space-filling behavior, we cannot be certain that all vortices or turbulent regions are truly captured by this measurement.

7. Conclusion

In this paper we proposed a method of filtering streamlines and identifying complex regions of the flow field. We described a method of quantifying flow field complexity by measuring the box counting ratio of streamlines seeded from the flow field. Using this box counting ratio, we are able to make a streamline complexity grid that has scalar values that represent the complexities of different regions. The streamline complexity grid along with the streamline box counting ratios allow for many visualization techniques such as filtering by local maximums and isosurfaces to allow the viewer to identify interesting parts of complex flow fields.

References

[BKH*15] BUJACK R., KASTEN J., HOTZ I., SCHEUERMANN G., HITZER E.: Moment invariants for 3D flow fields via normalization. In *2015 IEEE Pacific Visualization Symposium (PacificVis)* (apr 2015), Institute of Electrical & Electronics Engineers (IEEE). URL: http://dx.doi.org/10.1109/PACIFICVIS.2015.7156350, doi:10.1109/pacificvis.2015.7156350. 1

[CLSW14] CHAUDHURI A., LEE T.-Y., SHEN H.-W., WENGER R.: Exploring flow fields using space-filling analysis of streamlines. *Visualization and Computer Graphics, IEEE Transactions on 20*, 10 (Oct 2014), 1392–1404. doi:10.1109/TVCG.2014.2312009. 2

[HEWK03] HEIBERG E., EBBERS T., WIGSTROM L., KARLSSON M.: Three-dimensional flow characterization using vector pattern matching. *Visualization and Computer Graphics, IEEE Transactions on 9*, 3 (July 2003), 313–319. doi:10.1109/TVCG.2003.1207439. 2

[JBTS08] JÄNICKE H., BÖTTINGER M., TRICOCHE X., SCHEUERMANN G.: Automatic detection and visualization of distinctive structures in 3D unsteady multi-fields. *Computer Graphics Forum 27*, 3 (may 2008), 767–774. URL: http://dx.doi.org/10.1111/j.1467-8659.2008.01206.x, doi:10.1111/j.1467-8659.2008.01206.x. 1

[Jän10] JÄNICKE HEIKE; SCHEUERMANN G.: Towards automatic feature-based visualization, 2010. URL: http://dx.doi.org/10.4230/DFU.SciViz.2010.62, doi:10.4230/DFU.SciViz.2010.62. 1

[KW10] KHOURY M., WENGER R.: On the fractal dimension of isosurfaces. *Visualization and Computer Graphics, IEEE Transactions on 16*, 6 (Nov 2010), 1198–1205. doi:10.1109/TVCG.2010.182. 2

[LMSC11] LEE T.-Y., MISHCHENKO O., SHEN H.-W., CRAWFIS R.: View point evaluation and streamline filtering for flow visualization. In *Visualization Symposium (PacificVis), 2011 IEEE Pacific* (March 2011), pp. 83–90. doi:10.1109/PACIFICVIS.2011.5742376. 1

[MAD05] MEBARKI A., ALLIEZ P., DEVILLERS O.: Farthest point seeding for efficient placement of streamlines. In *Visualization, 2005. VIS 05. IEEE* (Oct 2005), pp. 479–486. doi:10.1109/VISUAL.2005.1532832. 1

[MBS*04] MAHROUS K., BENNETT J., SCHEUERMANN G., HAMANN B., JOY K.: Topological segmentation in three-dimensional vector fields. *Visualization and Computer Graphics, IEEE Transactions on 10*, 2 (March 2004), 198–205. doi:10.1109/TVCG.2004.1260771. 2

[MCHM10] MARCHESIN S., CHEN C.-K., HO C., MA K.-L.: View-dependent streamlines for 3d vector fields. *Visualization and Computer Graphics, IEEE Transactions on 16*, 6 (Nov 2010), 1578–1586. doi:10.1109/TVCG.2010.212. 1

[MHHI98] MAO X., HATANAKA Y., HIGASHIDA H., IMAMIYA A.: Image-guided streamline placement on curvilinear grid surfaces. In *Visualization '98. Proceedings* (Oct 1998), pp. 135–142. doi:10.1109/VISUAL.1998.745295. 1

[MJL*13] MCLOUGHLIN T., JONES M., LARAMEE R., MALKI R., MASTERS I., HANSEN C.: Similarity measures for enhancing interactive streamline seeding. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 19*, 8 (2013), 1342–1353. URL: http://www.sci.utah.edu/publications/Mcl2013a/McLoughlin_TVCG2013.pdf, doi:10.1109/TVCG.2012.150. 1

[SP99] SADARJOEN I. A., POST F. H.: Geometric methods for vortex extraction. In *Data Visualizationï€¡99* (1999), Springer, pp. 53–62. 2

[SPM*98] SADARJOEN I., POST F., MA B., BANKS D., PAGENDARM H.-G.: Selective visualization of vortices in hydrodynamic flows. In *Visualization '98. Proceedings* (Oct 1998), pp. 419–422. doi:10.1109/VISUAL.1998.745333. 2

[SS06] SALZBRUNN T., SCHEUERMANN G.: Streamline predicates. *Visualization and Computer Graphics, IEEE Transactions on 12*, 6 (Nov 2006), 1601–1612. doi:10.1109/TVCG.2006.104. 1

[TB96] TURK G., BANKS D.: Image-guided streamline placement. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1996), SIGGRAPH '96, ACM, pp. 453–460. URL: http://doi.acm.org/10.1145/237170.237285, doi:10.1145/237170.237285. 1

[XLS10] XU L., LEE T.-Y., SHEN H.-W.: An information-theoretic framework for flow visualization. *Visualization and Computer Graphics, IEEE Transactions on 16*, 6 (Nov 2010), 1216–1224. doi:10.1109/TVCG.2010.131. 1

[YKP05] YE X., KAO D., PANG A.: Strategy for seeding 3d streamlines. In *Visualization, 2005. VIS 05. IEEE* (Oct 2005), pp. 471–478. doi:10.1109/VISUAL.2005.1532831. 1

[ZHA98] ZHONG J., HUANG T., ADRIAN R.: Extracting 3d vortices in turbulent fluid flow. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 20*, 2 (Feb 1998), 193–199. doi:10.1109/34.659938. 2